1
Introdução

1.1
Motivação

A preocupação e os esforços empregados para melhorar as práticas de desenvolvimento de software buscando o aumento da produtividade e da qualidade, bem como a redução de custos e esforços, evidenciam novas perspectivas para o desenvolvimento de software.

A dificuldade e a demora na implementação de um sistema complexo podem ser reduzidas se forem utilizados componentes previamente desenvolvidos e testados. Contudo, é preciso um grande esforço inicial para que uma empresa de desenvolvimento de software crie um repositório de componentes reutilizáveis com este fim.

A engenharia de software tem procurado cada vez mais automatizar esse processo de modelagem e codificação. Hoje em dia, a programação orientada a objetos é largamente utilizada e, naturalmente, um esforço grande tem sido feito no sentido de reutilizar software. Recentemente, foi desenvolvida por Dandashi (1998) uma técnica para inferir a reutilização de um componente de software utilizando medidas retiradas diretamente da implementação. Simultaneamente, ferramentas de modelagem e geração automatizada de código, as ferramentas CASE, estão cada vez mais completas.

O paradigma de orientação a objetos mudou os elementos que são utilizados para inferir a qualidade do software. Métricas tradicionais de produtos de software como tamanho, complexidade, performance e qualidade tiveram que ser modificadas para contemplar novas noções como encapsulamento, herança, e polimorfismo que são inerentes a orientação a objetos. Com isso, novas métricas foram definidas (Chidanber & Kemerer, 1994) (Hitz & Montazeri, 1996) (Li & Henry, 1993) para medir os produtos da orientação a objetos.

Contudo, muitas das métricas e modelos de qualidade (Lorenz & Kidd, 1994) só podem ser aplicadas após o desenvolvimento completo ou semi-completo do software. Elas atuam na informação extraída da implementação do produto e oferecem informação muito tardia para ajudar a melhorar a qualidade interna do software antes que este esteja pronto. Assim, existe uma necessidade de

se utilizar métricas que atuem nas primeiras fases do desenvolvimento do software para garantir que a modelagem possui características favoráveis que levarão a um software final de maior qualidade.

Bansiya (2002) desenvolveu um modelo chamado Modelo Hierárquico de Qualidade para Modelagens Orientadas a Objetos (QMOOD), que procura mapear métricas de modelagem em atributos de qualidade. Esses atributos de qualidade são: Reutilização, Flexibilidade, Inteligibilidade, Funcionalidade, Extensibilidade e Efetividade. Essa pesquisa de Bansiya é uma das bases desta dissertação, e cada um desses atributos de qualidade será detalhado mais adiante.

A idéia de melhorar modelagens de software orientado a objetos baseado em métricas de qualidade deve ser um processo de automatização, criando modelagens de maior qualidade baseadas nas características do problema. Este processo de automatização sugere o uso de técnicas computacionais avançadas como a Inteligência Computacional.

A Inteligência Computacional busca, através de técnicas inspiradas na Natureza, o desenvolvimento de sistemas inteligentes que imitem aspectos do comportamento humano, tais como: aprendizado, percepção, raciocínio, evolução e adaptação. Algoritmos Genéticos são modelos inteligentes inspirados na evolução biológica que, através de métodos adaptativos, conseguem localizar soluções potenciais, sem considerar exaustivamente todas as possíveis soluções para o problema (Goldberg, 1989) (Davis, 1996) (Michalewicz, 1996) (Bäck et al, 1997).

Técnicas de Inteligência Computacional, principalmente Algoritmos Evolucionários, vêm sendo aplicadas em problemas de geração de código orientado a objetos há algum tempo. Em sua tese de doutorado, Bruce (1995) utilizou programação genética para automatizar o processo, abordando problemas mais específicos da geração do código em si e sua funcionalidade. Entretanto, apesar de mencionar a possibilidade, a evolução de códigos reutilizáveis não é abordada.

Por causa do aumento da demanda de produtos de software, a necessidade de um rápido desenvolvimento e alta qualidade em cada etapa do processo, proporcionando o reuso, torna-se clara. Por esse motivo, modelos que ajudam a automatizar e controlar a qualidade de cada etapa do desenvolvimento estão cada vez mais populares, e esta pesquisa investiga um dos caminhos possíveis.

1.2
Objetivos do Trabalho

O objetivo fundamental deste trabalho é estudar, criar e avaliar um modelo que integre a técnica de Algoritmos Genéticos Co-evolucionários a técnicas de engenharia de software orientado a objetos e às métricas propostas por Bansiya (2002) no seu modelo QMOOD (Modelo Hierárquico de Qualidade para Modelagens Orientadas a Objetos), para encontrar mecanismos de suporte ao design que permitam aumentar a qualidade do software. Nesse suporte, são considerados aspectos pertencentes ao paradigma de orientação a objetos: classes, dependências, métodos e atributos. Para se alcançar este objetivo, é preciso desenvolver uma representação genética para o problema e modelar uma função de avaliação que contemple as métricas de qualidade.

1.3
Descrição do Trabalho

Esta pesquisa foi estruturada através dos seguintes passos: estudar a área de Engenharia de Software orientado a objetos e métricas de qualidade; definir um modelo de algoritmo genético co-evolucionário para a síntese de modelagens de software orientado a objetos visando a melhoria da qualidade; implementar uma ferramenta baseada nesse modelo e avaliar o seu desempenho a partir de estudos de caso.

O estudo sobre a área de Engenharia de Software orientado a objetos e métricas de qualidade consistiu no levantamento de material sobre as fases de desenvolvimento de software e as diversas representações em UML mais utilizadas. O estudo das métricas de qualidade consistiu em identificar métricas mais relevantes e que atuassem no início do processo de desenvolvimento. Com isso, pode-se identificar e viabilizar a possibilidade de aplicar computação evolucionária para melhoria da qualidade do software.

A definição do modelo de algoritmo genético co-evolucionário cooperativo consistiu na definição de espécies, na representação do cromossomo de cada espécie, na especificação da função de avaliação e na definição dos operadores genéticos.

No modelo de algoritmo genético co-evolucionário cooperativo duas ou mais populações interagem de forma cooperativa de modo a se atingir um ou mais objetivos em comum. Existem duas razões principais para se decompor a solução do problema em diversas populações. Em primeiro lugar, os algoritmos genéticos convencionais impedem, a longo prazo, a preservação de certos componentes da solução pois, por estarem codificados por completo em um indivíduo, eles são avaliados como um todo e apenas os subcomponentes que pertencem a indivíduos com avaliações altas serão preservados. Em segundo lugar, o fato da representação estar relacionada a uma solução completa e por não haver interações entre os membros da população, não existe pressão evolucionária para a ocorrência de co-adaptação, ou seja, não existe pressão para a adaptação de um subcomponente dada a ocorrência de uma mudança em outro subcomponente (Cruz, 2003).

Para a definição do modelo foram criadas quatro espécies, que colaboram entre si durante a evolução. Cada espécie fica responsável por evoluir uma parte do problema, assim, o problema foi dividido em componentes a serem evoluídos. Estes são: classes, dependências, métodos e atributos. Todos esses são pertencentes ao paradigma de orientação a objetos.

A representação da espécie responsável pelas classes empregou um cromossomo baseado em ordem que leva em consideração a ordem em que as classes serão preenchidas. Nesta representação, o cromossomo é uma lista de classes, sendo que as posições no início do cromossomo têm maior prioridade e serão preenchidas antes das outras. Como o número de classes na modelagem pode variar, esse cromossomo deve possuir tamanho variável. Para isso, foi utilizada uma máscara binária para indicar qual posição está ou não ativa (Zebulum et al, 2001).

Ainda na espécie responsável pelas classes, a herança entre classes deve ser contemplada. Portanto, foi criada outro tipo de máscara, agora sendo representada por uma matriz binária.

A representação da espécie responsável pelas dependências entre classes empregou um cromossomo binário em duas dimensões e, com isso, torna-se um cromossomo binário clássico.

As espécies responsáveis pelos métodos e atributos têm representações idênticas e são baseadas na representação utilizada por Cruz (2003) em sua

pesquisa, onde os métodos e atributos podem ser vistos como os recursos a serem alocados. Essa representação foi bem testada por Cruz (2003), inclusive os operadores genéticos específicos para ela.

Os cromossomos de todas as espécies juntas, após a decodificação, representam metadados de um diagrama de classes UML (OMG, 2003) simplificado que, posteriormente, podem ser entrada de uma ferramenta CASE, o qual pode construir o código da implementação propriamente dito, no padrão escolhido. Uma descrição gráfica do processo proposto é mostrada na Figura 1.

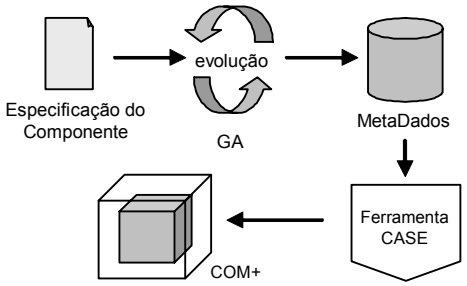

Figura 1 – Descrição do processo de automatização do desenvolvimento do software

A função de avaliação, por sua vez, foi definida objetivando a síntese de uma modelagem de software orientado a objetos com uma maior qualidade. Neste problema, deseja-se contemplar mais de um objetivo ao mesmo tempo. Foram considerados seis métricas como objetivo: Reutilização, Flexibilidade, Inteligibilidade, Funcionalidade, Extensibilidade e Efetividade. Para isso, foi utilizada a técnica de Pareto para problemas multi-objetivos (Deb et al, 2000) (Deb, 2001). Utilizando a pesquisa feita por Bansiya (2002), a evolução está direcionada no sentido da melhoria das métricas de qualidade, e com isso é esperado que a modelagem sintetizada seja qualitativamente melhor.

Os operadores genéticos (crossover e mutação), por sua vez, buscam recombinar o valor dos genes, de modo a obter novas modelagens com melhor qualidade (no caso, a qualidade é representada pelos seis objetivos da função de avaliação). Para isto, foram utilizados para os cromossomos, operadores binários, máscaras binárias e operadores baseados em ordem, como os utilizados por Cruz (2003), que são comumente utilizados em problemas baseados em ordem.

Finalmente, a implementação de uma ferramenta que representasse o modelo foi essencial para os testes e avaliação. A ferramenta foi implementada

utilizando o GACOM (*Genetic Algorithms Components*), um *Framework* para modelagem de algoritmos genéticos (Guimarães, 2004). Foram realizados estudos de caso para validar a funcionalidade do modelo. As modelagens sintetizadas foram comparadas com modelagens consideradas de boa qualidade por especialistas para se avaliar a melhoria completa ou parcial da modelagem.

1.4 Trabalhos Relacionados

Esta área de pesquisa se mostrou ainda muito inexplorada, assim, as pesquisas relacionadas a este trabalho são poucas, e a maioria utiliza a técnica de Programação Genética.

A Programação Genética (PG), concebida por John Koza (Koza, 1992) no início dos anos 90, apresenta um mecanismo de funcionamento semelhante ao de algoritmos genéticos. Entretanto, ao invés de utilizar palavras binárias para representar possíveis soluções, estruturas de dados em forma de *árvores* de tamanho arbitrário são empregadas.

Em Programação Genética tem-se uma população de programas de computador, e o objetivo do algoritmo é, através da evolução destes programas, chegar a um programa que melhor resolva um determinado problema.

Um dos trabalhos destacados é a realizada por Blickle (1996) onde é inspecionada a aplicação de Algoritmos Evolucionários na síntese de sistemas digitais complexos compostos por elementos de *hardware* e *software*. Em sua pesquisa, Blickle (1996) mostra a síntese de implementações para um chip de *codec* de vídeo H. 261, demonstrando a metodologia e a capacidade dos Algoritmos Evolucionários obterem boas soluções para o problema.

Ao longo dos anos, a Programação Genética teve que ser adaptada para seguir cada vez mais o paradigma da Orientação a Objetos. Um dos principais trabalhos em Programação Genética Orientada a Objetos (OOGP) foi realizado por Bruce (1995), mas esta área ainda é considerada uma área pouco explorada e pesquisada. Bruce (1995) aborda problemas mais específicos da geração do código em si e sua funcionalidade, e apesar de mencionar, não aborda a evolução de códigos reutilizáveis.

Pesquisas mais recentes estão sendo realizadas, como o artigo de Lucas (2004) onde são consideradas técnicas mais avançadas da Orientação a Objetos na

Programação Genética. Nesse artigo, Lucas ainda sugere o possível uso de Métricas de Software para avaliar programas sintetizados.

Apesar de existirem alguns trabalhos que procuram automatizar sistemas de software utilizando técnicas de Inteligência Computacional, o assunto abordado nesta pesquisa ainda procura um novo caminho onde não existem muitas bases para comparação. Isso em parte aumenta as dificuldades encontradas, não fornecendo nenhum tipo de benchmarks ou direções a serem seguidas.

1.5 Organização da Dissertação

Esta dissertação está dividida em cinco capítulos adicionais, descritos a seguir.

O capítulo 2 resume o levantamento sobre as etapas do desenvolvimento de software orientado a objetos, detalha em particular as métricas de qualidade utilizadas para a avaliação das modelagens.

O capítulo 3 discute a técnica de Algoritmos Evolucionários, detalhando os modelos co-evolucionários cooperativos e estratégias de avaliação multi-objetivo.

O capítulo 4 descreve em detalhes o modelo desenvolvido nesta dissertação, apresentando as representações utilizadas neste trabalho, a decodificação das espécies e a função de avaliação elaborada.

O capítulo 5 descreve a implementação de uma ferramenta genérica para o modelo e os estudos de caso testados e resultados.

Por fim, o capítulo 6 apresenta as conclusões do trabalho e sugere novas direções para a continuação da pesquisa apresentada.